III. AMENDMENT TO THE CLAIMS:

This listing of claims will replace all prior versions, and listings of claims in the application:

Listing of Claims:

1. (Currently Amended) A computerized method for scheduling execution of one or more processes in a computer, comprising the steps of:

providing a timer;

providing a series of events for each process of a preselected set of processes, the events comprising a start time for each process;

~~starting execution of each process based on a time out of the timer, each process starting execution according to the corresponding start time~~;

providing a table;

storing each event in the table; and

operating the timer to be set for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next event in the table;

executing the plurality of processes on the computer based upon the series of events, each process starting execution based on the timeout of its respective time out event on the timer.

2. (Original) The method according to claim 1 wherein the events further include a deadline for each process of the preselected set of processes;

based on a time out of the timer, stopping execution of an executing process regardless of whether the process has stopped execution normally, each process stopping execution according to the corresponding deadline.

3. . (Original) The method of claim 1, wherein if one process starts execution while another process is executing, preempting the process already executing.

4.. (Original) The method of claim 1 wherein each process comprises one of a task and an ISR.

5.. (Original) The method of claim 4 wherein when a process comprises an ISR, upon execution of the ISR, providing an enable time.

6.. (Original) The method of claim 5 comprising the further step of disabling the ISR after execution; and enabling the ISR upon expiration of the enable time.

7. (Currently Amended) A computer system comprising:
a computer including:

a timer;

a table storing a series of events for each process of a preselected set of processes, the events comprising a start time for each process;

an operating system executing on the computer and causing execution of each process based on a time out of the timer, each process starting execution according to the corresponding start time stored in the table, the timer being arranged and configured to be set for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next event in the table.

8. (Original)The computer system of claim 7 wherein the table further stores events comprising a deadline for each process, the operating system causing, based on a time out of the timer, execution of an executing process to stop regardless of whether the process has stopped execution normally, each process stopping execution according to the corresponding deadline stored in the table.

9. (Original) The computer system of claim 8, wherein each process comprises one of a task and an ISR.

10. (Currently Amended) The computer system of claim 9 wherein when one of the set of processes ~~a process~~ comprises an ISR, wherein after ~~upon~~ execution of the ISR based on the timeout of its start time, the ISR is disabled for a predetermined period of time, and then enabled upon expiration of the predetermined period of time ~~using an enable time in the table, the enable time causing disablement of an ISR after execution; and enablement of the ISR upon expiration of the enable time~~.

11. (Original) A method for scheduling one or more processes comprising the steps of:

providing a timer;

starting a plurality of processes based on a time out of the timer, each process starting execution according to a start time specified in a time table;

if one of the processes starts execution while another process is executing, preempting the process already executing;

if one of the processes has been preempted and the process that preempted the process stops execution, resuming the process that has been preempted; and

based on a time out of the timer, stopping execution of the processes regardless of whether the process has stopped execution normally, each process stopping execution according to a deadline specified in the time table;

setting the timer for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next one of a start time and deadline in the time table.

12. (Original)A method for scheduling one or more processes comprising the steps of:

providing a plurality of timers;

starting a plurality of processes based on a time out of a first one of the timers, each process starting execution according to a start time specified in a time table; and

based on a time out of a second one of the timers, stopping execution of the processes regardless of whether the process has stopped execution normally, each process stopping execution according to a deadline specified in the time table;

setting the first one of the timers for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next start time in the time table;

setting the second one of the timers for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next deadline in the time table.

13. (Currently Amended) The method according to claim 12 wherein each process comprises one of a task and an ISR, ~~and~~ the method comprising the further steps of :

when one of the processes~~a process~~ comprises an ISR, ~~upon execution of the ISR,~~ providing an enable time; and

~~, disabling the ISR after execution; and enabling the ISR upon expiration of the enable time, the step of enabling being performed by~~

upon beginning execution of the ISR based upon the timeout of the first timer based on the start time of the ISR, setting a third one of the timers to time out after a reload value equal to the enable time, and disabling further instances of the ISR until the time out of the third timer.

14. (Currently Amended)A computer system comprising:
a computer comprising:

a timer ~~mechanism~~;

a table storing a series of events for each process of a preselected set of processes, the events comprising a start time for each process and a deadline for each process;

an operating system executing on the computer and causing execution of each process based on a time out of the timer ~~mechanism~~, each process starting execution according to the corresponding start time stored in the table and stopping execution according to the corresponding deadline stored in the table, the timer ~~mechanism~~ being arranged and configured to be set for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next event in the table;

the timer ~~mechanism~~ comprising a first timer set for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next start time in the time table, and a second timer set for a time out at each of a series of reload values, each reload value being equal to a number of time increments until a next deadline in the time table.

15. (Original) The computer system of claim 14 wherein each process comprises one of a task and an ISR, and further comprising a third timer to time out after a reload value equal to an enable time wherein the enable time is set, when a process comprises an ISR, upon execution of the ISR, the ISR being disabled after execution; and enabled upon time out of the third timer at the expiration of the enable time.

16. (new) A computerized method for executing of one or more processes in a computer, comprising the steps of:

assigning a plurality of sequential start times for each of a plurality of processes;

assigning a deadline time for each of the plurality of processes, each deadline time representing a latest stop time its respective process;

starting execution of a first of the plurality of processes on the computer at its respective start time, and setting a timer to a first value at the respective start time, the first value being equal to a number of time increments from the first start time until a start time of a next start time of the plurality of sequential start times;

upon expiration of the timer, stopping execution of the first process and starting execution of the next process;